What to expect of a 30 y.o. SCOTCH

MUMPS User Days 2023

F. Pellegrini
INRIA Bordeaux – Sud-Ouest

Sommaire

Inria

01

The SCOTCH project

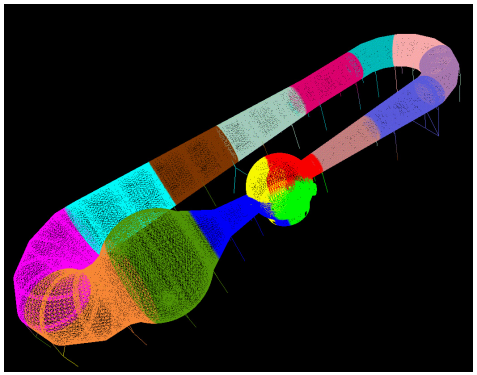
Inria

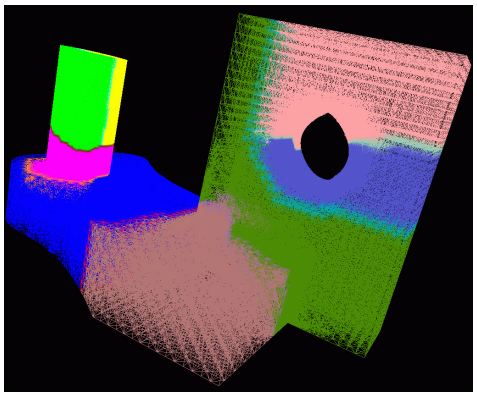
The SCOTCH project

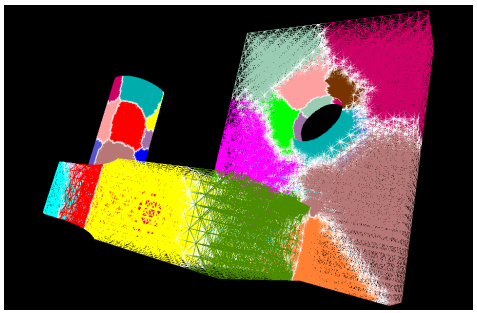
- Started 01 December 1992 (now 30 y.o.!)
  - Currently in v7.0.3
- Tackles the graph partitioning and mapping problems by way of algorithms that rely only on graph topology
  - Geometry is never taken into account
  - Hierarchical description of target architectures
    - Can also map onto parts of a regular architecture
- Provides a software toolbox:
  - SCOTCH centralized software library and tools
    - POSIX pThreads
  - PT-SCOTCH distributed-memory software library and tools
    - MPI + POSIX pThreads
  - Compatibility library for replacing METIS & PARMETIS in existing software

Inria

- Two main problems are considered:
  - Domain decomposition for iterative methods
  - Sparse matrix ordering for direct methods
- These problems can be modeled as graph partitioning problems on the adjacency graph of symmetric positive-definite matrices
  - Edge separator problem for domain decomposition
  - Vertex separator problem for sparse matrix ordering by nested dissection
  - Also: partitioning with overlap







Inria

- Stands for "Parallel and Threaded Scotch"
  - > Offspring (and part) of the Scotch project
- Devise robust parallel graph partitioning methods
  - > Meant to handle graphs of more than a billion vertices distributed across more than a thousand processors
  - > Improve over sequential graph partitioning methods if possible
    - Devise new algorithms
    - Provide new features to existing algorithms, such as fixed vertices, multi-weight algorithms, etc.
  - > Provide graph repartitioning and remapping methods

Inria

02

Scotch v7.0

Inria

- Latest major revision of SCOTCH
- Full ascending compatibility (as always!)
- New major features:
  - Dynamic multi-threading
  - Full reentrance
  - Improved reproducibility
- New environment features:
  - CMake support
  - Improved METIS & PARMETIS support

Inria

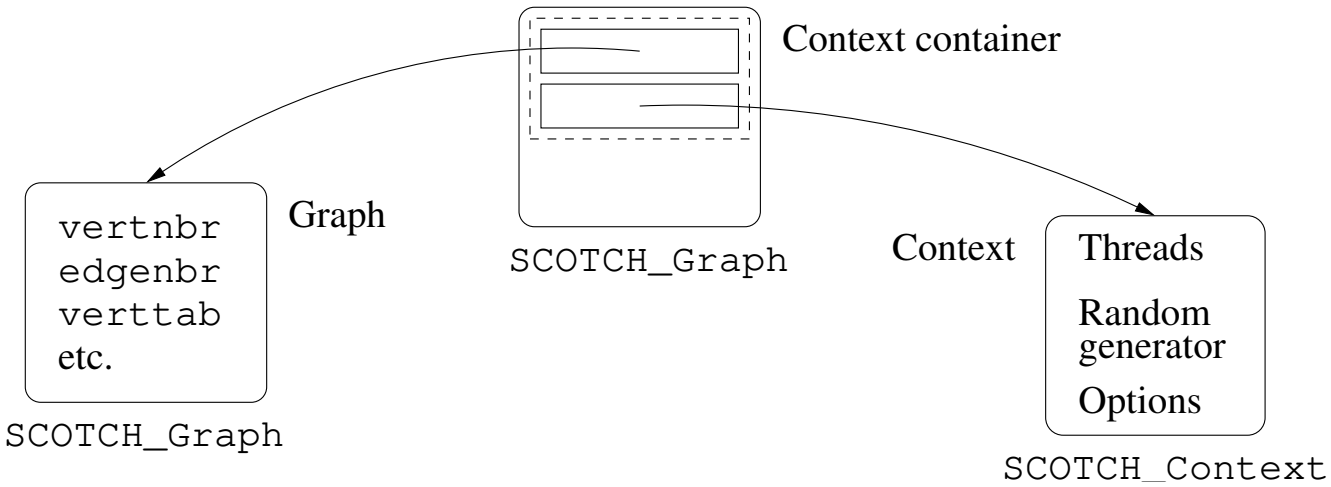
Challenges of multi-threading

- Use all "available" threads whenever possible
  - > Depending on the user's will
  - > Import existing user's threads if needed
- Run partitioning tasks concurrently
  - > Software must be fully reentrant
  - > Fine control on reproducibility
    - Control on (pseudo-)randomness
    - Control on thread concurrency (deterministic multi-threaded algorithms)

Design choices

- Handle threads manually
  - > No "foreign" and rigid top-down thread management like OpenMP
  - > Use raw POSIX pThreads
    - – Allows to capture existing threads
    - – Allows to assign sub-pools of threads to specific tasks
- Handle random generator(s) manually
  - > Lighweight Mersenne twisters
  - > Allows to assign a generator to each task and thread
- Provide deterministic variants of non-deterministic threaded algorithms
  - > Dynamic selection

- Embed all run-time information for running a SCOTCH task
  - > Attached pool of threads
  - > Pseudo-random benerator and seed
  - > Configuration options
- Encapsulated in opaque `Container` objects
  - > New opaque objects that pretend to be SCOTCH traditional objects such as `Graph`, `Dgraph` and `Mesh`
  - > No visible change to the SCOTCH interface!



Areas for concurrency

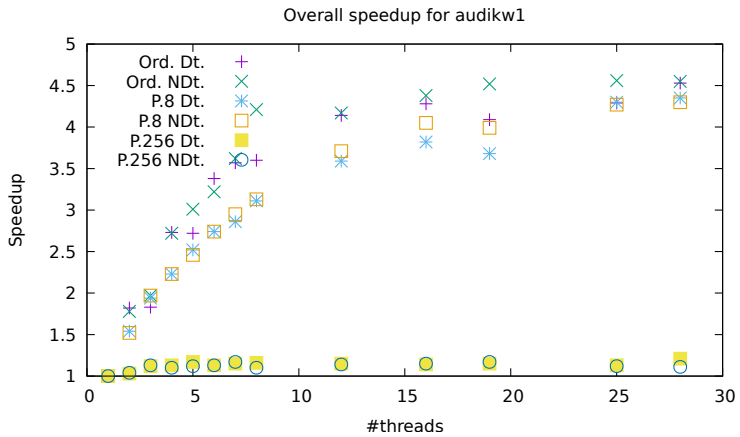
- Recursive bipartitioning & nested dissection frameworks
  - > Embarrassingly parallel
  - > No fine-grain control on load balancing
  - > Requires to split in two a pool of threads
  - > Requires a thread-safe MPI implementation for PT-Scotch
- Multilevel frameworks
  - > Vertex matching
    - – Can we preserve a deterministic behavior?
  - > Construction of the coarsened graph
    - – Use of non-compact graphs
- Diffusion-based algorithms
  - > Very good candidates but very expensive as well
    - – About 40 times more than Fiduccia-Mattheyses

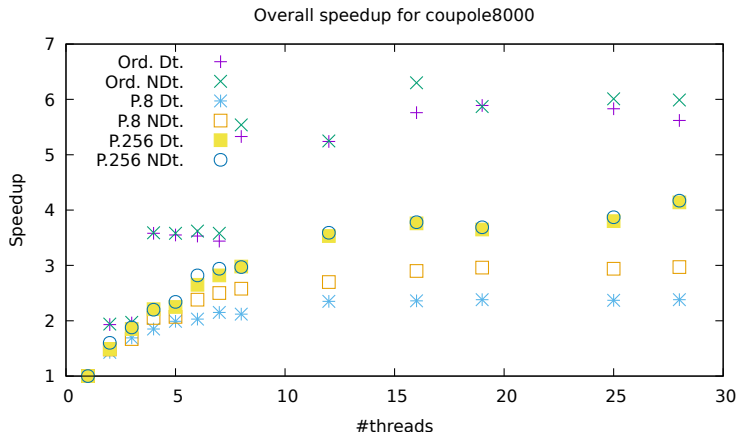
Inria

- Multiprocessor: dual-processeur Intel Xeon Broadwell E5-2650L v4 @1.7 GHz, 2 $\times$ 14 cores, 64 GiB main memory
- Test graphs:

Name	$\|V\|$	$\|E\|$	$\Sigma\delta/\|V\|$
audikw1	943695	38354076	26.66
coupole8000	1768161	41656975	10.94

Early results





- For partitioning/mapping, when k-way band graphs cannot be created, run time is dominated by sequential local optimization algorithms (e.g. Fiduccia-Mattheyses)
  - > Need to split FM computations across areas

Critical review

- Graph partitioning is essentially a memory-bound problem
  - > Quasi-linear-time heuristics are available for many kinds of common graphs (meshes, etc.)
  - > Algorithms that parallelize well are much more expensive (e.g. diffusion-based algorithms, genetic algorithms)
    - – Yet they do not scale enough to fully replace their sequential counterparts in current settings
- Multi-threaded algorithms in Scotch v7.0 are however useful in production contexts
  - > Make sure data placement is not "too expensive" compared to subsequent computations
  - > The new thread model gives much more flexibility to users

Inria

Early user feedback

- Users may not want to use multi-threaded MPI implementations
  - > Can degrade performance by up to 20% for some communication-intensive user codes
- Packagers configure PT-Scotch with multi-threaded features activated
  - > I encouraged them to...
- Parametrization should be dynamic (like many others):
  - > Deterministic behavior, memory/speed trade-off (non-compact graphs), etc.
- Issue of high priority
  - > As early as Scotch v7.0.5

Inria

03

Roadmap & Consortium

Inria

- v7.0.4 almost ready for roll-out
  - > Bugfix release
  - > One more fix and go!
- v7.0.5 should arrive soon
  - > More fixes
  - > More dynamic parametrization
  - > Re-engineered multi-threaded genetic algorithm
  - > Should be the "production-grade" reference version for branch v7.0

Inria

- Vertex separation framework that balances the size of the halos between the two parts ($\mathrm{v}7.1$)
- Multi-constraint partitioning ($\mathrm{v}8.0$)
- Parallel static mapping and dynamic remapping ($\mathrm{v}?$)
- Etc.

- Aims at gathering organizations that want to secure the future of SCOTCH as an essential toolbox for their activities:
  - Maintenance
  - New developments
- Call for founding members is on-going
- All the relevant information is here:

`https://team.inria.fr/tadaam/call-for-founding-members-for-the-scotch-consortium/`